



esProc

Innovated data
computing engine

High Performance Computing Database (SPL Base)

Issued by Raqsoft

What's esProc?



High Performance Computing Database - SPL Base



Offline Batch Running



Online Computing



Multidimensional
analysis



What does esProc solve?



Batch running is slow

Batch running in the middle of the night can't be finished, and there is no time for mistakes. It's even more frightening at the end of month or beginning of year.



Response is slow

Operation of correlation statistics is slow, and interface drag is slow. Pre-aggregation scheme occupies too much space and has too many functional blind areas.



Query is slow

Have to wait for 10 minutes for the query result of a report, and the business personnel is very angry. Unable to get query result in case of too many people or the time span is too long.



Cost is high

Spending a lot of money on the memory database, and the performance is still not good enough.

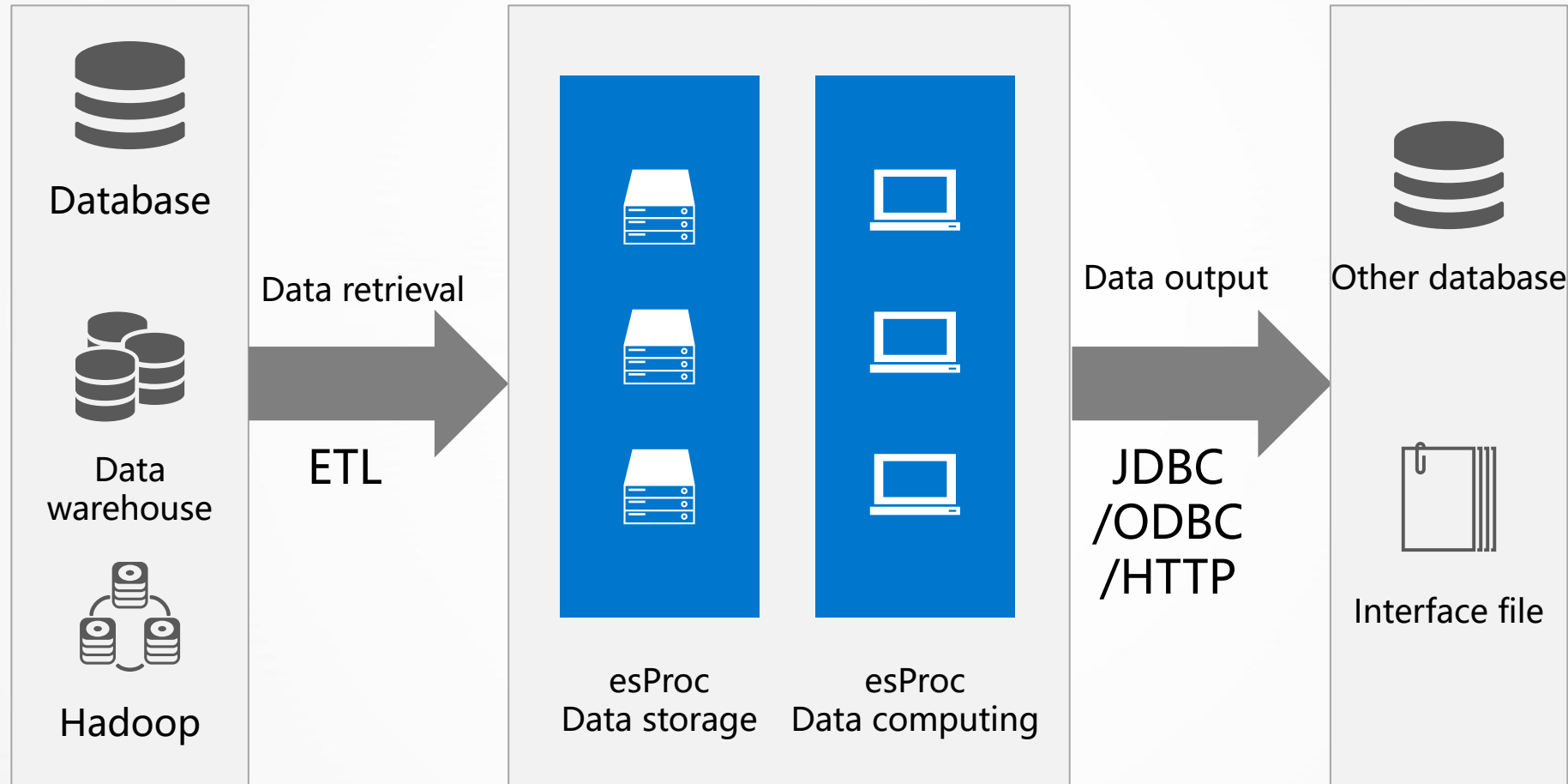
No matter single machine or cluster, no matter international brand or domestic rookie, no matter MPP or HADOOP, the average performance increase by esProc is several times to tens of times!

* Understanding esProc Applicable Scenarios: <http://c.raqsoft.com/article/1567394256889>

Offline batch running



[scenario characteristics] No concurrency, no need for real-time, huge amount of data, high requirement for time window

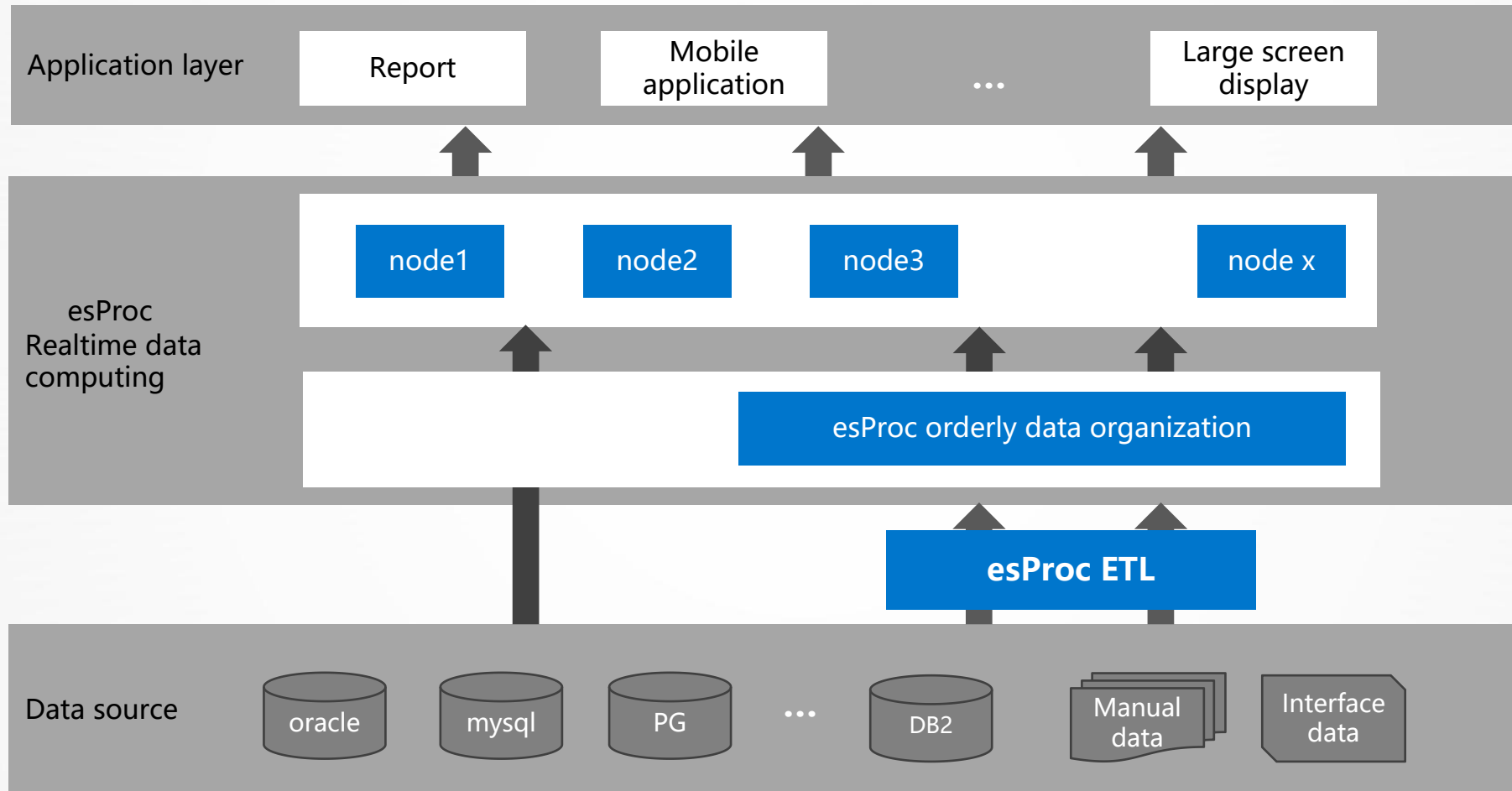


*For details: <http://c.raqsoft.com/article/1567393607177>

Online query



【 scenario characteristics 】 Multi-concurrency, business may be complex, second-level response, large amount of data needs cluster support

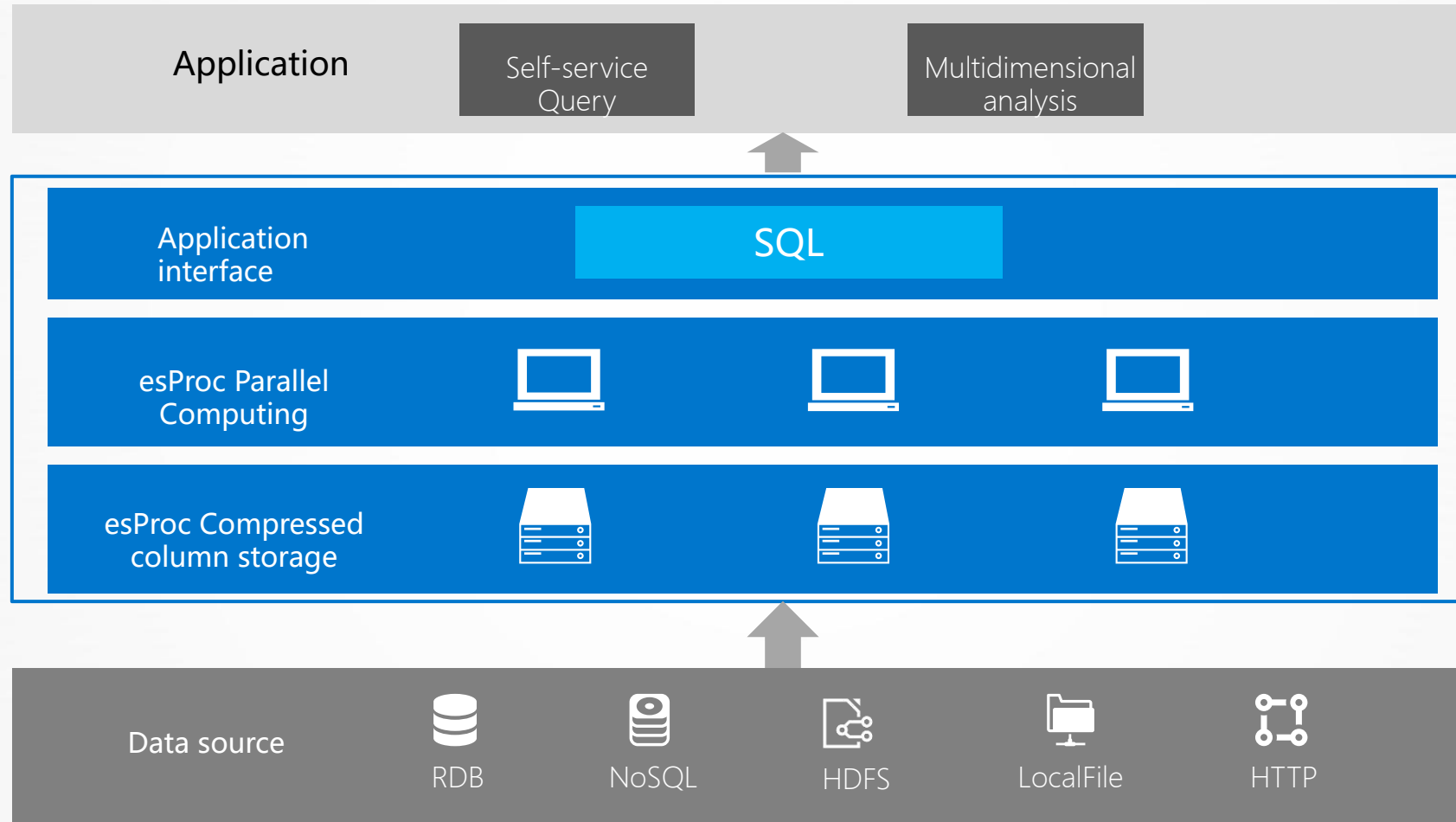


*For details: <http://c.raqsoft.com/article/1567392671017>

Multidimensional analysis



【 scenario characteristics 】 Multi-concurrency, real-time response, relatively regular calculation, support for general BI tools



*For details: <http://c.raqsoft.com/article/1567391781827>



Performance optimization case 1

Performance optimization of historical insurance policy associated business batch running in insurance industry

Complex rules

What is the same car? The frame number is the same, the vehicle VIN number is the same, the license plate number and the type are the same. Whether Vehicles are on loan, commercial insurance and traffic compulsory insurance, etc.

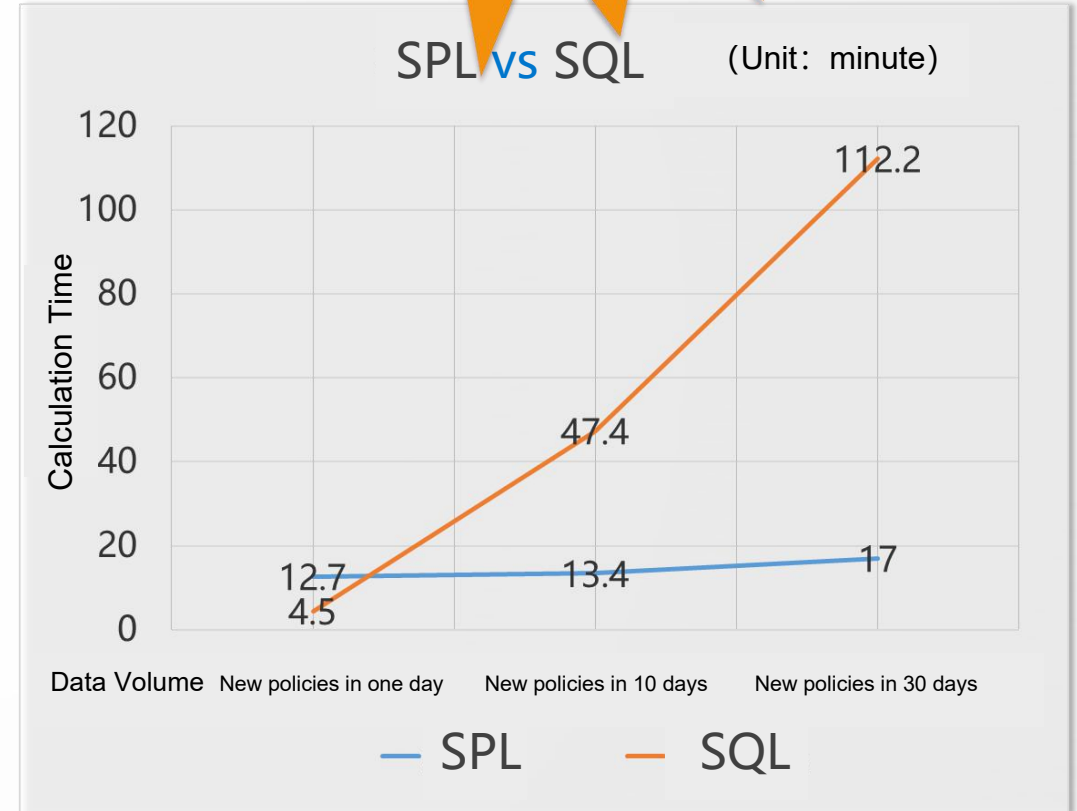
Large amount of data

Hundreds of millions of data in a province

Long batch running time

New policy within 30 days take 2 hours, and new policy within 90 days take a long time without any result.

Facing the Current Situation



Optimizing effect

Performance optimization case 2



Solution of Large Concurrent Real-time Query for Massive Accounts



Huge data volume, more than 300 million rows

From September 2015 to September 2018

High number of visitors concurrently

Hundreds of thousands and millions of visitors

Query time should not exceed 1 second

Using 6 ES servers to basically meet the response requirements

But code table associations can no longer be implemented

It takes hours to regenerate data when the code table changes



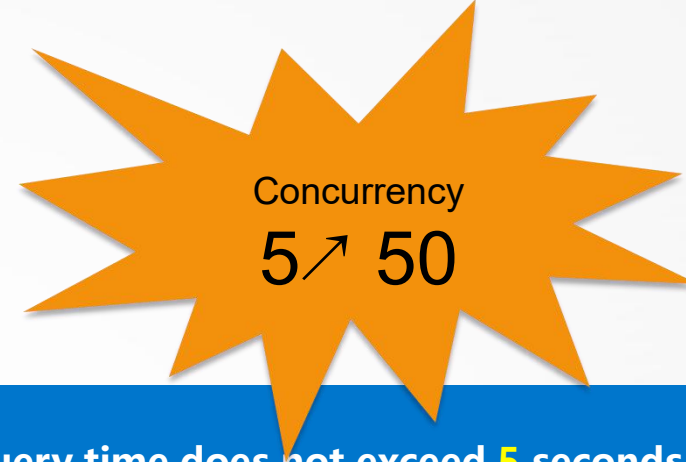
Facing the Current Situation

esProc: Same performance with a single server, implementing code association at the same time.



Performance optimization case 3

A Speed-up Scheme for Multi-user Self-service Analysis with Large Data Volume in banking industry



Self-service Analysis System - High Performance Requirements

Hundreds of users access and expect second-level response
Expect to query the full amount of data

Data Warehouse - Uncontrollable Performance

Undertaking too much application load, can only open five connections for self-service analysis system, performance is greatly affected by other applications.

Facing the Current Situation

Query time does not exceed 5 seconds

Comparable to professional column-storage data warehouse

Each server supports 50 concurrencies

Actual support for hundreds of user access without pressure

More than 30 million pieces of data

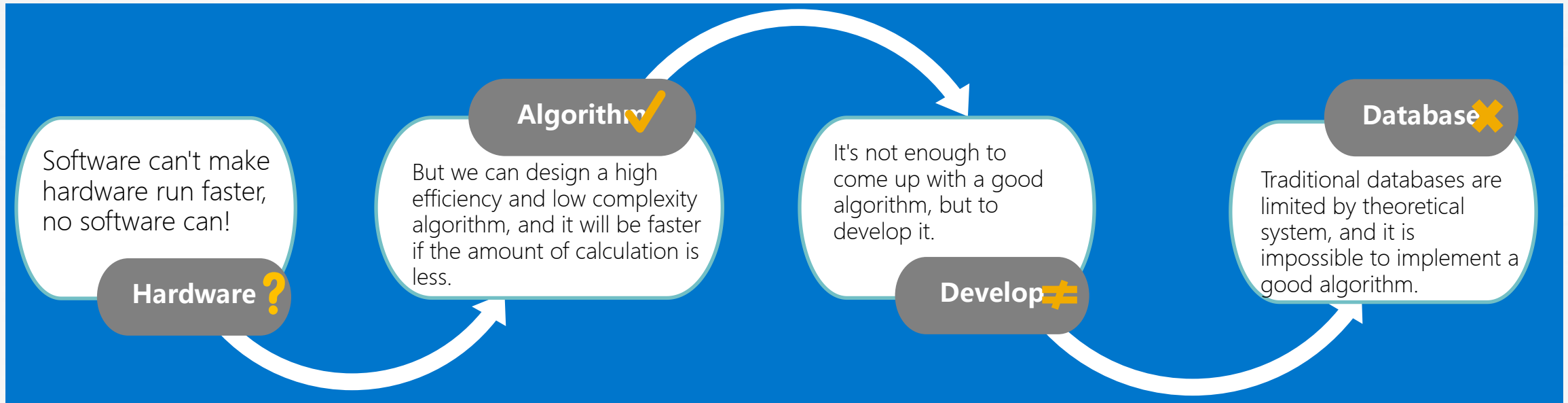
Filter and search for results according to conditions

Fully compatible with self-service analysis tools

Only need to change the JDBC configuration to esProc

esProc application effect

esProc High Performance Computing Concept



What can we do? → Look forward!

Oh, so it is like this. → Yes, that's not magical.

Then find a programmer to do it. → Not so easy!

Isn't that all you can do is stare in despair? → Hey hey, that's how it works most of the time.





High performance of esProc comes from innovative computing system

[analogy] Calculate $1+2+3+\dots+100=?$

Ordinary people will do like this

$1+2=3$
 $3+3=6$
 $6+4=10$
 $10+5=15$
 $15+6=21$
 $21+7=28$
...

Gauss does like this

$1+100=101$
 $2+99=101$
...
A total of fifty 101
 $50*101=5050$

Gauss was smart enough to think of efficient algorithms, but more importantly, people had invented **multiplication** at that time.





Question: How to get the top 10 of 100 million rows of data in SQL?

In theory, SQL will get the top 10 after complete sorting, which is inefficient. Programmers know that there is a way to achieve this operation without complete sorting, but they can not express it in SQL. They can only count on the database engine to optimize automatically, but the database will not optimize in complex situations.

SQL in relational databases is like an arithmetic system with only **additions**, while SPL in esProc invents **multiplication**! esProc also have more multiplications (high-performance computing and storage libraries), and everyone can be Gauss (fast implementation of high-performance algorithms).

Part of High Performance Computing Mechanism Provided by esProc



 Traversal technique	 Highly efficient Joins	 High performance storage	 Distributed computing
Delayed cursor	Foreign key as pointer	Orderly Compressed Storage	Preemptive Load Balancing
Aggregate Understanding	Numbering of foreign keys	Free column storage	Fork-reduce
Ordered cursor	Order-based merge	Hierarchical Numbering positioning	External storage redundancy fault tolerance
Traversal reuse	Integration of Main sub table and same dimension table	Flake Index and Caching	Memory spare tire fault tolerance
Prefilter traversal	Unilateral HASH Join	Double increment segmentation	Cluster dimension table

Many of the algorithms here are original inventions of esProc!

Examples of High Performance Algorithms



	A	
1	<code>=file("data.ctx").create().cursor()</code>	
2	<code>=A1.groups(;top(10,amount))</code>	Orders in the Top 10
3	<code>=A1.groups(area;top(10,amount))</code>	Top 10 orders per region
converting high-complexity sorting to low-complexity aggregation		

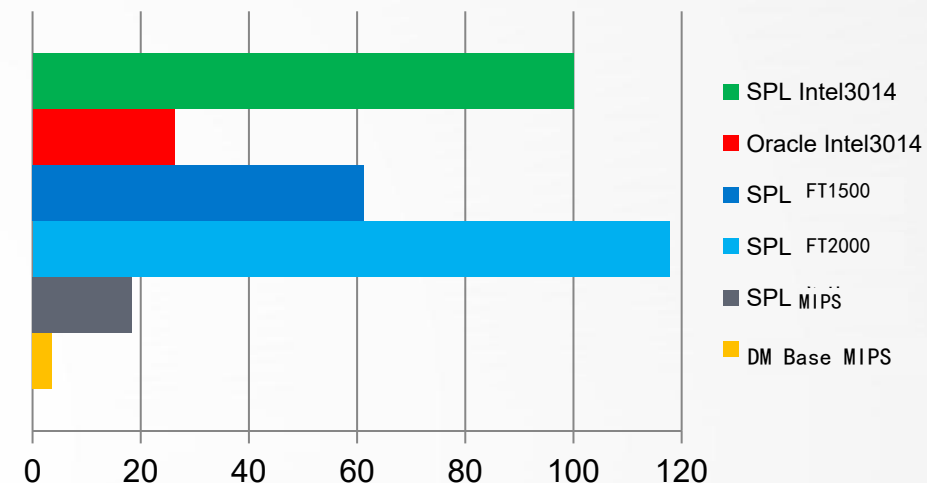
	A	
1	<code>=file("order.ctx").create().cursor()</code>	Prepare for traversal
2	<code>=channel(A1).groups(product;count(1):N)</code>	Configure reuse computing
3	<code>=A1.groups(area;sum(amount):amount)</code>	Traverse and get grouped results
4	<code>=A2.result()</code>	Get the result of the reuse computing
multiple result sets can be returned by one traversal		

Performance of esProc

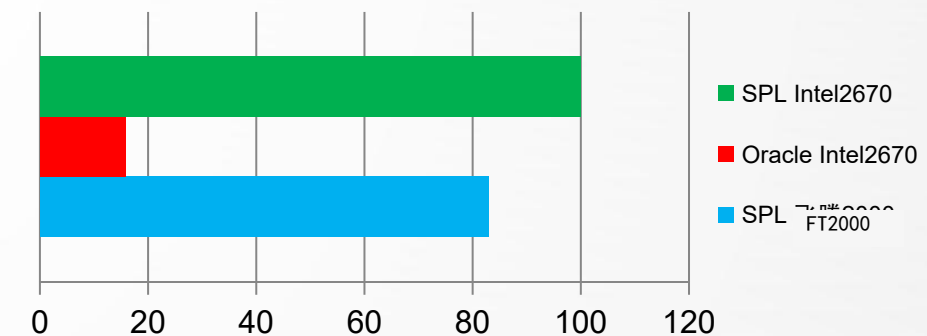


TPCH (unit: second)	100G						200G		
	Intel 12 core		ARM16 core	ARM64core	MIPS 8 core		Intel 16 core		ARM64 core
	SPL	Oracle	SPL	SPL	SPL	DM Base	SPL	Oracle	SPL
1	38	131	62	19	275	507	54	325	36
2	4	27	8	6	29	247	9	73	13
3	40	222	62	44	215	4451	47	582	94
4	24	207	43	18	230	1790	29	454	44
5	40	225	76	28	337	1761	47	463	54
6	10	135	22	7	63	757	12	352	13
7	50	184	75	37	235	700	45	496	70
8	55	192	115	59	386	1611	71	485	144
9	81	234	125	68	552	1066	91	636	144
10	29	215	66	23	164	1634	56	493	50
11	7	33	12	8	52	165	11	63	13
12	25	184	72	38	181	647	57	464	55
13	63	37	158	110	339	2209	255	103	252
14	38	157	65	12	143	500	65	368	43
15	34	155	60	31	109	506	68	358	50
16	11	13	21	15	66	105	36	71	28
17	29	165	52	16	189	963	42	349	32
18	21	344	36	13	194	2382	29	966	28
19	41	154	69	12	137	518	59	345	37
20	37	175	61	13	112	594	59	442	25
21	233	326	228	190	936	3349	204	790	398
22	25	48	39	27	129	139	41	99	49
合计	935	3563	1527	794	5073	26601	1387	8777	1672

TPCH 100G



TPCH 200G



Intel3014 1.7G/12 core/64G memory ARM16 FT1500/16 core/32Gmemory MIPS/8core/64G memory Intel2670 2.6G/16core/128G memory ARM64 FT2000/64core/256G memory

Performance of esProc



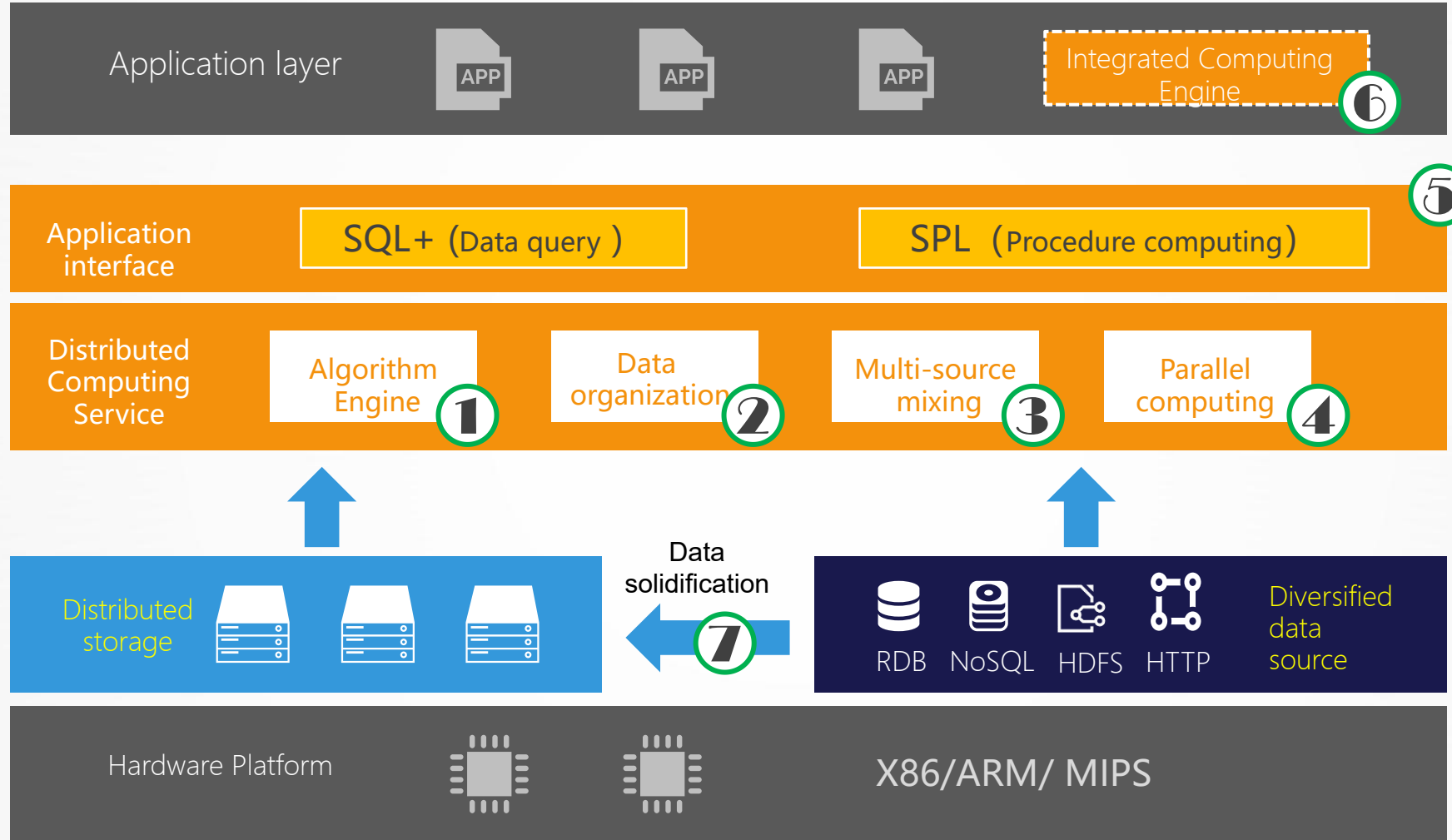
* Data size: 10 billion rows; cluster size: 4

Testing results (Time unit: second)					
Testing cases	Intel X86 chip			Domestic Feiteng chip	
	SPL Reading File Computing	SPL Reading Database Computing	SQL Computing in Database	SPL Reading File Computing	SPL Reading Database Computing
Multi-to-multi Join traversal	69	103	>1 hour	93	268
Ordered grouping traversal	100	647	>1 hour	102	2037
Multistep procedure calculation	272	848	>1 hour	377	>1 hour
Grouping	39	155	2573	56	2493
Large table Join grouping	111	566	>1 hour	178	2106
Batch key value query	15	>1 hour	>1 hour	15	>1 hour

【Note】 **SPL** is the programming language used in the Raqsoft esProc; **SQL** is the programming language used in relational database.

Running Raqsoft esProc on domestic Feiteng chip can surpass the performance of distributed database on Intel chip.

esProc SPL Base Architecture



High Performance Storage



High Performance Data Storage

Private data storage format: set files, group tables

File System Storage Form

Support to store data by business classification in tree directory

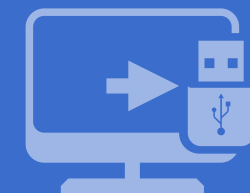
Set files

Double increment segmentation supports arbitrary number parallelism
Self-owned efficient compression technique (reduced space; less CPU usage; secure)
Generic Storage, allowing set data



Group tables

Mixed row and column storage
Ordered storage improves compression rate and positioning performance
Efficient intelligent index
Double increment segmentation supports arbitrary number parallelism
Integration of main and sub table to reduce storage and association
Numbering keys to achieve efficient positioning Join

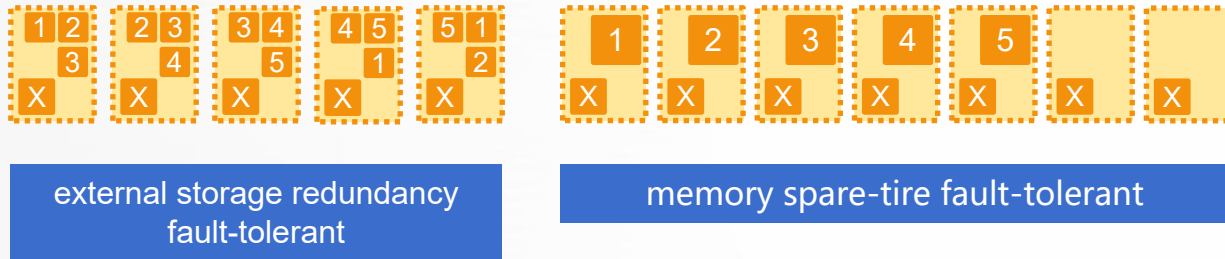


Distributed Computing



Data Fault Tolerance and Computational Fault Tolerance

Provide two kinds of data fault-tolerant mechanisms, external storage redundancy fault-tolerant, memory spare-tire fault-tolerant



Support computing fault tolerance, automatically migrate the computing task of the node to other nodes to continue to complete when the node fails.

Controllable Data Distribution

Users can flexibly customize data distribution and redundancy schemes according to the characteristics of data and computing tasks, effectively reduce the amount of data transmission between nodes, in order to achieve higher performance.

Centerless Architecture to Avoid Single Point Failure

Clusters do not have permanent central master nodes, and programmers use code to control the nodes involved in computing.

Load Balancing Capability

Decide whether to assign tasks according to the degree of idleness (number of threads) of each node to achieve an effective balance of load and resources.

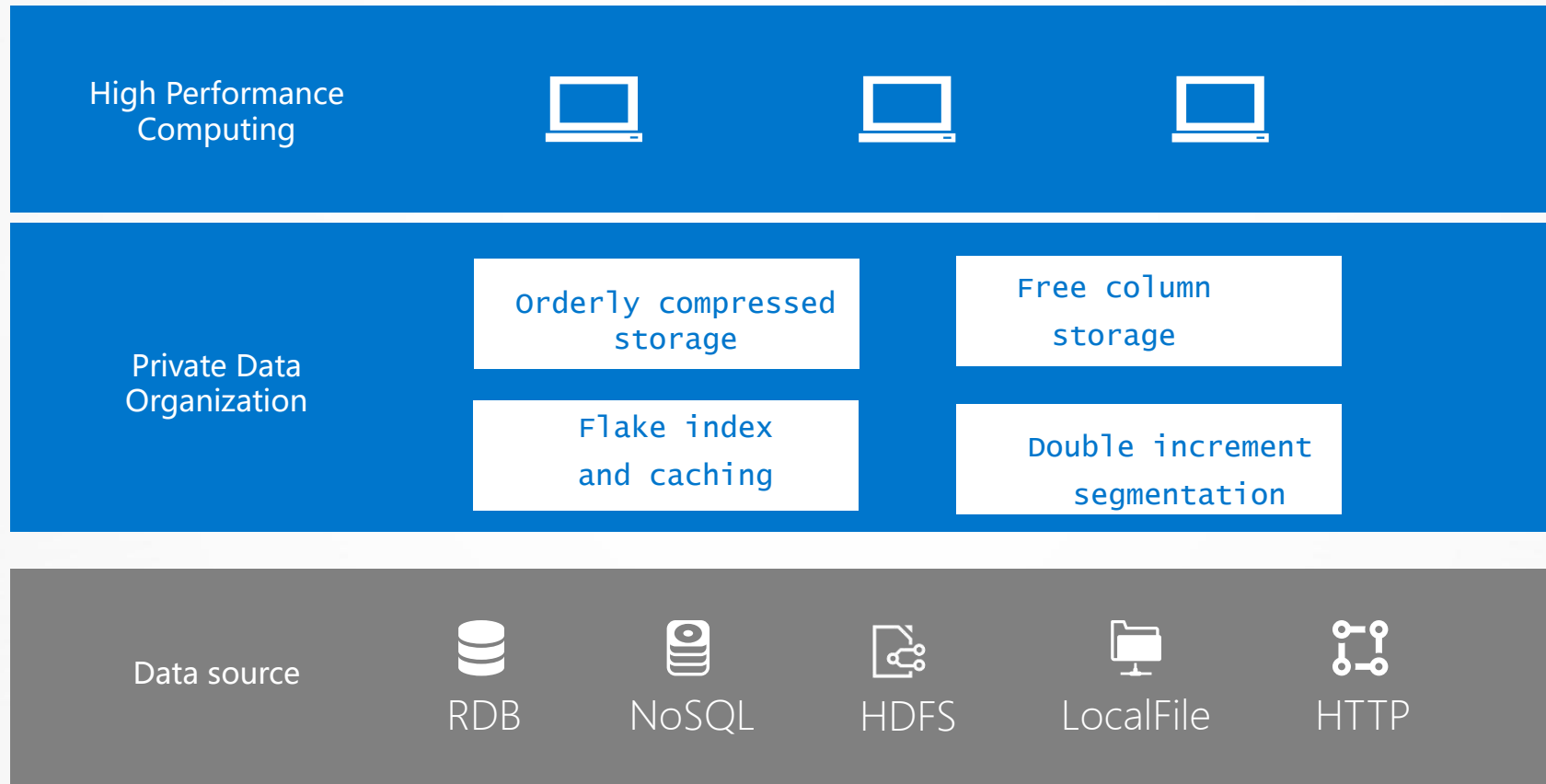
FAQ

Frequently asked
questions



Does esProc store data by itself?

It must! The storage of data-intensive computing is the guarantee of performance. The inefficient storage of traditional RDB and HADOOP can not achieve high performance.

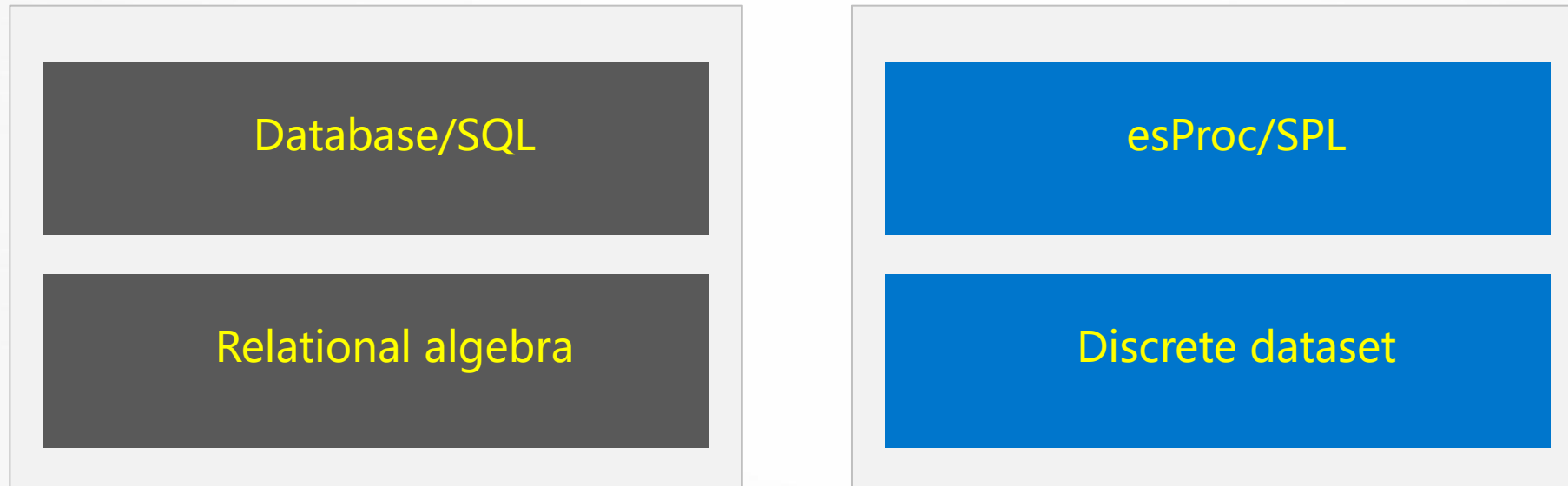


esProc designs special and efficient data organization schemes for [memory](#), [external storage](#) and [cluster](#), which are suitable for a variety of computing scenarios.



Is esProc based on open source or database technology?

esProc is based on a [brand-new computing model](#), no open source technology can be cited, and all independent innovation from theory to code.



esProc based on innovation theory can no longer use SQL to achieve high performance, and SQL can not describe most low complexity algorithms.

Only provides high performance SQL interface for multidimensional analysis with regular operation form to adapt to various front-end BI tools.



How difficult is it to learn esProc?

esProc is dedicated to performance optimization and provides a dedicated SPL syntax

Learning SPL is not difficult. It can be mastered in hours and skilled in weeks.

What's difficult is to design optimization algorithms!



To learn SPL is as easy as turning over the palm



Optimizing algorithm is much more difficult to design

So we designed the following optimization process



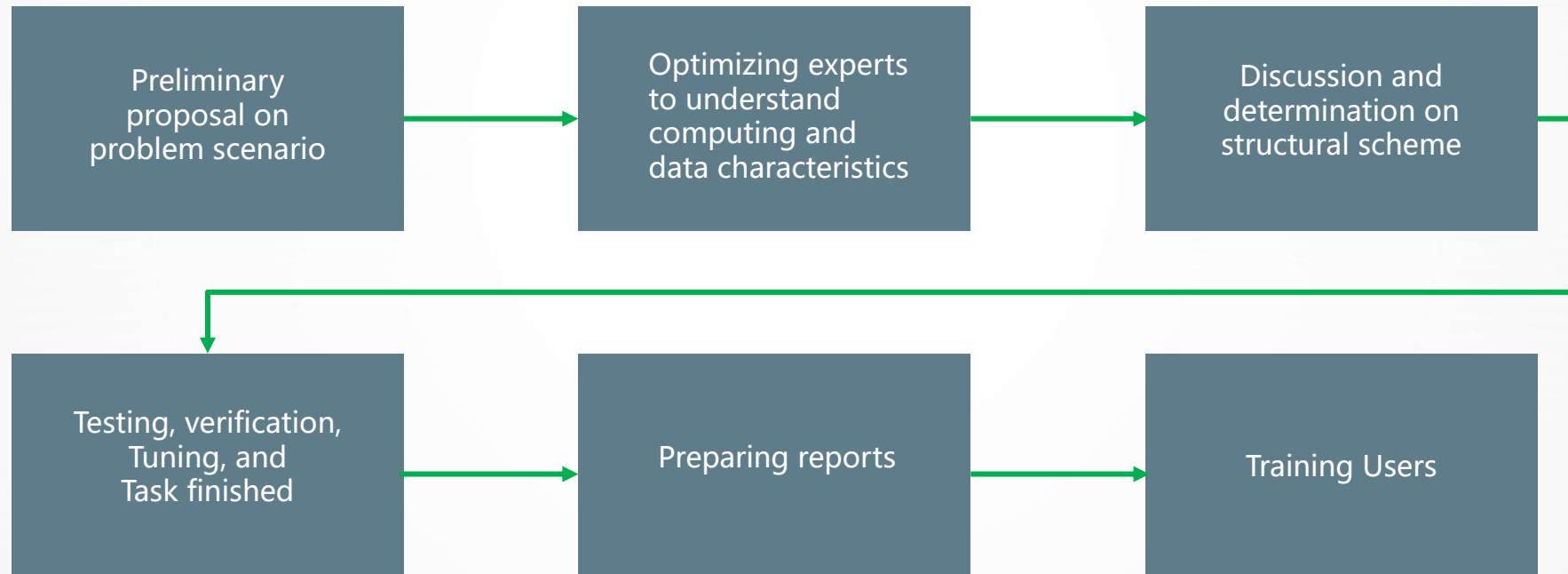
Performance optimization process



The first 1-2 scenarios will be implemented by Raqsoft senior engineer in collaboration with users.

Most programmers are used to the way of thinking in SQL and are not familiar with high performance algorithms. They need to be trained to understand in one or two scenarios.

Dozens of performance optimization routines will be experienced and learned. Algorithmic design and implementation are not so difficult.



Give a man a fish and you feed him for a day. Teach him how to fish and you feed him for a lifetime!

Innovation makes progress!

